

REMARKS

Reconsideration of the application as amended is respectfully requested. Claims 506-609 are currently pending, and have been rejected by the Examiner.

In the accompanying amendment, claim 599 has been amended to correct minor typographical errors therein. It is respectfully submitted that the amendments to claim 599, do not add new matter, and are therefore suitable for entry.

Specification

The Examiner objected to the specification. In particular, the Examiner has stated that:

“The abstract and the specification of the disclosure are objected to because the use of the trademark term “Java” has been noted in the abstract and the specification. It should be capitalized each letters in the word, or include a proper trademark symbol, such as TM or © wherever it appears and be accompanied by the generic terminology. Appropriate correction is required.”

In response, the applicants have amended the Abstract, and the specification to clearly indicate that the term “Java” is a registered trademark. Accordingly, it is respectfully submitted that the Examiner should withdraw her objection to the specification.

Drawings

The Examiner has objected to the drawings because Figures 7A-7D were not described in the “Brief Description of Drawings” section of the specification. In response, the Applicants have amended their specification to include a brief description of Figures 7A-7D. Accordingly, it is respectfully submitted that the Examiner should withdraw her rejection of the drawings.

Claim rejections under 35 U.S.C. § 112.

The Examiner has rejected claims 508, 521, 541, 557, 561, 575, 608, and 609 under 35 U.S.C. § 112, second paragraph. In particular, the Examiner has stated that:

“Claims 508, 521, 541, 557, 561, 575, 608 and 609 contain the trademark term “Java”. Where a trademark name is used in a claim as a limitation to identify or describe a particular material or product, the claim does not comply with the requirements of 35 U.S.C. § 112, second paragraph. See Ex parte Simpson, 218 USPQ 1020 (Bd. App. 182). The claim scope is uncertain since

the trademark or trade name cannot be used properly to identify a source of goods, and the goods themselves. Thus, a trademark or trade name does not identify or describe the goods associated with the trademark or trade name. In the present case, the trademark/trade name is used to identify/describe bytecodes and, accordingly, the identification/description is indefinite.”

Applicants however respectfully disagree with the Examiner in this regard. Firstly, the Applicants note that 35 U.S.C. § 112, second paragraph does not per se prohibit the use of a trademark in a claim. Instead, 35 U.S.C. § 112, second paragraph merely prohibits the use of a trademark in a claim, in circumstances where such use renders the claim indefinite. A claim is definite if the metes and bounds of the invention can be adequately determined from the claim language (in the Goffe, 526 F. 2d 1393, 188 USPQ 131 (CCPA 1975)). In Ex parte Simpson, 218 USPQ 1020 (Bd.App.1982) the use of the trademark “Hypalon” in the claims was found to have rendered the claims indefinite since, on the one hand, the claim language could have been narrowly construed to a particular chlorosulphonated ethylene having a specific group of additives employed by the owner of the “Hypalon” trademark to produce the desired properties, or on the other hand the claim language could have been broadly construed to encompass every synthetic resin. Thus, claim language was indefinite as to whether chlorosulphonated polythene was required to be present before infringement could occur.

However, in the present case the use of the trademark term “Java” in the claims does not render the claims indefinite. Apart from being a trademark, the term “Java” is also a programming language (see the attached definition of Java which was obtained from the website www.whatis.com). Since Java is also a particular programming language, it is impossible to refer to that programming language, or aspects of the programming language without using the term “Java”. Moreover, the use of the term “Java” in the claims does not create uncertainty as to the metes and bounds of the invention, as was the case with the use of the term “Hypalon” in the Ex parte Simpson case. Therefore, it is respectfully submitted, that the use of the trademark term “Java” in the claims does not render the claims indefinite under 35 U.S.C. § 112, second

paragraph. Accordingly, it is respectfully submitted that the Examiner should withdraw her rejection under 35 U.S.C. § 112, second paragraph.

Double Patenting

Examiner has rejected claims 509-609 under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claims 1-70 of the US Patent No. 6332215. In this regard, the Examiner has stated that “although the conflicting claims are not identical, they are not patentably distinct from each other because they are obvious variation (sic) each other”. Section 804 of the MPEP specifically states that any obviousness-type double patenting rejection should make clear:

- (A) The differences between the inventions defined by the conflicting claims – a claim in the patent compared to a claim in the application; and
- (B) The reasons why a person of ordinary skill in the art would conclude that the invention defined in the claim in issue is an obvious variation of the invention defined in a claim in the patent.

Applicants respectfully point out to the Examiner, that the Examiner has merely stated a conclusion that the conflicting claims are obvious, but has not articulated the differences between the inventions defined by the conflicting claims, and the reasons why a person of ordinary skill in the art would conclude that the invention defined in the claim in issue is an obvious variation of the invention defined in a claim in the patent, as required by the MPEP. Thus the Examiner has thus failed to establish a prima facie case of obviousness type double patenting, and accordingly, save for asserting that claims 509-609 are not obvious in view of the claims of U.S. Patent No. 6332,215, the Applicants are unable to respond to the Examiner’s arguments in this regard, with any greater particularity.

Claim Rejections Under 35 U.S.C. § 103

The Examiner has rejected claims 506-520, 522-550, 559-573, 575-598 under 35 U.S.C. § 103(a) as being unpatentable over Dickol et al (US 5,875,336), in view of the Krall et al article. With regard to claims 506 and 559, the Examiner has stated that:

- “maintaining data for register-based instruction... memory (Dickol abstract, col. 5, lines 13-20, Fig. 3, col. 3, lines 42-60)
- executing the stack-based instruction... instructions (Fig. 5, col. 4, lines 42-50).

Dickol does not specifically disclose overflow/underflow mechanism. However, Krall discloses the overflow/underflow mechanism (Krall, page 1021, lines 1-10). The modification would be obvious because one of the ordinary skill in the art would be motivated to resume normal operations of the stack to translate the non native code to a set of native codes efficiently.

Dickol does not specifically disclose that the system generates exception. However, Krall discloses the accelerator produces an exceptions (Krall, page 1026, lines 27-36, page 1027, line 1-8). The modification would be obvious because one of the ordinary skill in the art would be motivated to deal with errors (or exceptions) efficiently as they arise during running a program.”
(Page 4, Office Action mailed 2/19/2004)

Applicants however disagree with the Examiner that claims 506, and 559 are rendered obvious in view of the combination of the Dickol and Krall. In the “Background of the Invention” section, Dickol states that there is two commonly used methods to execute a non-native instruction set within a computer. (See col. 1, lines 60-65). The first method is to utilize a software interpreter, and the second method is to compile the program that is in non-native instructions to a set of native instructions. Dickol specifically points out the drawbacks of the second method in that this method “requires a large amount of memory for storing the compiler as well as the output of the compiler.” (See col. 2, lines 1-10). Dickol also states that in the second method, if the application is not available until it is invoked, the initial compilation time may add a significant delay before the execution begins. Dickol then concludes that both methods to execute a non-native instruction set within a computer inevitably result in performance degradation (see col. 2, lines 5-10). Dickol also states, at col. 2, lines 9-20:

“A better method would require a hardware element that translates non-native instructions to native instructions in “real time.” The term “real time” in this case means that the native instructions are

generated from the non-native instructions as soon as the processor within the computer attempts to perform an instruction fetched from a system memory. The present disclosure describes such a hardware element for efficiently performing the translation of non-native Java™ instructions to instructions that are native to the processor within the computer.”

In short, Dickol teaches a hardware technique to execute a non-native instruction set within a computer, and specifically teaches away from a compiler-based method to execute the non-native instruction set. Krall, on the other hand, teaches a compiler-based method for executing instructions of a non-native instruction set. For example, see the “Summary” section of Krall, where it is stated that “This paper describes the design and implementation of CACAO, a just-in-time compiler for Java™.” In fact, in the “Summary” section, Krall admits that the CACAO compiler suffers from some of the problems associated with the second method mentioned by Dickol. For example, Krall states that “Even though the CACAO system has to incur loading and compilation time, it executes Java™ programs up to 85 times faster than the JDK interpreter.”

Based on the foregoing, it will be seen that the teachings of Dickol, cannot be combined with the teachings of Krall, since Krall teaches a compiler-based technique for executing instructions of a non-native instruction set, whereas Dickol while teaching a hardware element that translates non-native instructions to native instructions in “real time” specifically teaches a way from a compiler based technique such as is disclosed in Krall.

Moreover, the Examiner’s statement that “Krall discloses the overflow/underflow mechanism (Krall, page 1021, lines 1-10), is incorrect. Krall, on page 1021, lines 1-10, describes the architecture of a RISC processor. Accordingly, it is not seen how the Examiner interprets statements made by Krall on page 1021, lines 1-10, as providing support for an overflow/underflow mechanism.

Based on the foregoing, it is respectfully submitted that the Examiner has failed to establish a prima facie case of obviousness. Accordingly, it is respectfully submitted, that the Examiner should withdraw her rejection of claims 506, and 559.

Given that claims 507-529, and claims 560-581 depend on claims 506, and 559, respectively, it is respectfully submitted that these claims are also not obvious in view of the combination of Dickol and Krall.

In rejecting independent claim 530, as being obvious in view of the combination of Dickol, and Krall, the Examiner merely states:

“Regarding claim 550 (Dickol, Abstract, Fig. 3, Fig. 4, Col. 3, lines 42-60, Col. 4, lines 50-60, Col. 5, lines 40-55), and (Krall, Page 1026, lines 27-36, Page 1027, lines 1-8)”.

Applicants traverse the Examiner’s rejection of claim 530, by pointing out to the Examiner that Dickol, by the Examiner’s own admission is deficient in that it does not “specifically disclose an overflow/underflow mechanism”, and that Krall fails to teach or suggest an overflow/underflow mechanism. The portions of Krall cited by the Examiner relate to exception handling, and not to an overflow/underflow mechanism. As will be noted by the Examiner, one limitation of claim 530 includes:

“Maintaining an operand stack associated with the instructions of the first instruction set in a register file including moving at least some operands between the register file and memory via at least one of an overflow and underflow mechanism”.

Thus, since the combination of Dickol, and Krall fails to teach or suggest an overflow/underflow mechanism, it is respectfully submitted that the combination of Dickol and Krall does not render claim 530 obvious. Moreover, and as argued above, Dickol cannot be properly combined with Krall, since Dickol specifically teaches away from Krall.

Given that claims 531-539 depend on claim 530, it is respectfully submitted that these claims are also not obvious in view of the combination of Dickol and Krall.

In rejecting claim 540, in view of the combination of Dickol and Krall, the Examiner has stated that:

“Regarding claim 540 (Dickol, Abstract, col. 3, lines 41-55, col. 4, lines 41-50).

Dickol does not specifically disclose the hardware processor is an accelerator. However, Krall discloses accelerator during translation is shown in Krall, summary lines 8-9. The modification would be obvious because one of the ordinary skill in the art would be motivated to compile, executes and run the program faster than the regular hardware accelerator.

Krall discloses virtual machine. (Krall, page 1017, Introduction). The modification would be obvious because one of the ordinary skill in the art would be motivated to compile and run the program in platform independent environment.

Krall discloses the accelerator produces an exceptions (Krall, page 1026, lines 27-36, page 1027, lines 1-8). The

modification would be obvious because one of the ordinary skill in the art would be motivated to deal with errors (or exceptions) efficiently as they arise during running a program.”
(Page 7, Office Action mailed 2/19/2004)

Applicants respectfully traverse the Examiner’s rejection of claim 540 by repeating the above argument, that Dickol may not be combined with Krall, since Dickol specifically teaches away from Krall. Moreover, the exception handling taught by Krall is based on states such as a reference to a null pointer, array index out of bounds, or division by zero. Krall fails to teach or suggest “generating an exception in respect of a selected stack-based instruction while in the accelerator mode”, as recited in claim 540. Thus, apart from failing to show that the Dickol and Krall references may be properly combined, the Examiner has also failed to show that the combination of Dickol and Krall teaches or suggests all limitations of claim 540. Accordingly, it is respectfully submitted that claim 540 is not rendered obvious by the combination of Dickol and Krall.

Given that claims 541-542 depend on claim 540, it is respectfully submitted that these claims are also not rendered obvious by a combination of Dickol and Krall.

In rejecting claim 543, in view of the combination of Dickol and Krall, the Examiner has stated that:

“Regarding claim 543, (Dickol, abstract, col. 3, lines 40-50, col. 4, lines 45-60, col. 6, lines 1-2), neither Dickol nor Krall disclose including program counter. Official notice is taken including program counter. The modification would be obvious because one of the ordinary skill in the art would be motivated to control the address of the next program sequence, to improve execution frequency.”
(Page 8, Office Action mailed 2/19/2003)

Applicants traverse the Examiner’s rejection of claim 543, by repeating the above argument that the Dickol reference may not be properly combined with the Krall reference, since Dickol specifically teaches away from Krall. Moreover, the Examiner’s argument with regard to claim 543 is not entirely clear. For example, the Examiner appears to be taking official notice by stating “official notice is taken including program counter (sic)”. However, it is not entirely clear what exactly the Examiner is taking official notice of. Bearing in mind, that claim 543 recites

“using a program counter for the plurality of instruction sets, the common program counter being stored in a common register”, it is difficult for the Applicant to determine how the official notice taken by the Examiner is related to the above quoted limitation of claim 543.

Based on the foregoing, it is respectfully submitted that the Examiner has failed to make a prima facie case that claim 543 is obvious in view of the combination of Dickol, and Krall. Accordingly, the Examiner is respectfully requested to withdraw her rejection of claim 543.

Given that claims 544-550 depend on claim 543, it is respectfully submitted that the Examiner should also withdraw her rejection of these claims.

Claim 583 includes the limitation of “at least one of an overflow and underflow mechanism to cause the operands to move between the register file and memory”. The Examiner’s rejection of claim 583 is based on the combination of Dickol, and Krall. Accordingly, the Applicants traverse the Examiner’s rejection of claim 583, in view of the foregoing arguments, that Dickol may not be properly combined with Krall, since Dickol teaches away from Krall. Further, even if Dickol may be properly combined with Krall, the combination still fails to teach or suggest an overflow/underflow mechanism, as recited in claim 583. Based on the foregoing, it is respectfully submitted that claim 583 is not rendered obvious by the combination of Dickol, and Krall.

Given that claims 584-591 depend on claim 583, it is respectfully submitted that these claims are also not rendered obvious in view of the combination of Dickol and Krall.

Applicants traverse the Examiner's rejection of claims 592, and 593 by repeating the above argument that Dickol may not properly be combined by Krall, since Dickol teaches away from Krall.

Moreover, as regards claim 592, this claim includes the limitation of “generating an exception in respect of a selected stat based instruction while in the accelerator mode”, which limitation is not taught or suggested by the combination of Dickol and Krall, Krall merely teaching an exception handling technique based on state rather than on a specific instruction. Thus, even if Dickol may properly be combined with Krall, the combination does not teach or

suggest all limitations of claim 592, and therefore for this additional reason, it is respectfully submitted that claim 592 is not rendered obvious by Dickol.

Regarding claim 593, it will be noted that this claim includes the limitation of "an overflow and underflow mechanism", which as argued above is not taught or suggested by Krall. Thus, even if Dickol may be properly combined with Krall, the combination fails to teach or suggest all limitations of claim 593. Accordingly, it is respectfully submitted that claim 593 is not rendered obvious by the combination of Dickol and Krall.

Given that claims 594-598 depend on claim 593, it is respectfully submitted that these claims are also not rendered obvious by the combination of Dickol and Krall.

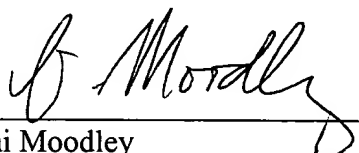
It is respectfully submitted that in view of the amendments and remarks set forth herein, all rejections have been overcome. All pending claims are now in condition for allowance, which is earnestly solicited.

Authorization is hereby given to charge our Deposit Account 02-2666 for any charges that may be due. Furthermore, if an extension is required, then Applicant hereby requests such an extension.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN

Dated: 5/4/, 2004



Vani Moodley
Limited Recognition Under 37 CFR § 10.9(b)

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, CA 90025-1030
(408) 720-8300